



ゲスト シェル

ゲストシェルは仮想化された Linux ベースの環境で、Python などのカスタム Linux アプリケーションを実行して Cisco デバイスを自動で制御および管理するために設計されています。システムの自動プロビジョニング（デイゼロ）も含まれます。このコンテナシェルは、ホストデバイスから分離された安全な環境を提供します。ユーザはそこで、スクリプトまたはソフトウェア パッケージをインストールし、実行することができます。

このモジュールでは、ゲストシェルとそれを有効にする方法について説明します。

- [ゲストシェルについて](#) (1 ページ)
- [ゲストシェルを有効にする方法](#) (11 ページ)
- [ゲストシェルの設定例](#) (18 ページ)
- [ゲストシェルに関するその他の参考資料](#) (22 ページ)
- [ゲストシェルの機能情報](#) (22 ページ)

ゲスト シェルについて

ゲスト シェルの概要

ゲストシェルは、仮想化された Linux ベースの環境であり、Cisco デバイスの自動制御と管理のための Python アプリケーションを含む、カスタム Linux アプリケーションを実行するように設計されています。ゲストシェルを使用して、サードパーティ製 Linux アプリケーションをインストール、更新、および操作することもできます。ゲストシェルはシステムイメージとともにバンドルされており、Cisco IOS コマンド `guestshell enable` を使用してインストールできます。

ゲストシェル環境は、ネットワーキングではなく、ツール、Linux ユーティリティ、および管理性を意図したものです。

ゲストシェルは、ホスト（Cisco スイッチおよびルータ）システムとカーネルを共有します。ユーザーは、ゲストシェルの Linux シェルにアクセスし、コンテナの `rootfs` にあるスクリプトおよびソフトウェア パッケージを更新することができます。ただし、ゲストシェル内のユーザーは、ホストのファイルシステムおよびプロセスを変更することはできません。

ゲストシェル コンテナは、IOx を使用して管理されます。IOx は、Cisco IOS XE デバイスのためのシスコのアプリケーション ホスティング インフラストラクチャです。IOx は、シスコ、パートナー、およびサードパーティの開発者によって開発されたアプリケーションおよびサービスをネットワーク エッジデバイスでシームレスにホスティングすることを、各種の多様なハードウェアプラットフォームにおいて可能にします。

次の表は、ゲストシェルのさまざまな機能とサポート対象のプラットフォームに関する情報を提供します。

表 1: Cisco ゲストシェルの機能

	ゲストシェル Lite (限定的な LXC コンテナ)	ゲストシェル (LXC コンテナ)
オペレーティング システム	Cisco IOS XE	Cisco IOS XE
サポートされるプラットフォーム	<ul style="list-style-type: none"> • Cisco Catalyst 3650 シリーズ スイッチ (全モデル) • Cisco Catalyst 3850 シリーズ スイッチ (全モデル) 	<ul style="list-style-type: none"> • Cisco ISR 4000 シリーズ サービス統合型ルータ (最低 8 GB の RAM を有するモデル)
ゲスト シェル環境	Montavista CGE7	CentOS 7
Python 2.7	サポート対象 (Python V2.7.11)	サポート対象 (Python V2.7.5)
カスタムの Python ライブラリ	<ul style="list-style-type: none"> • Cisco 組込イベント マネージャ • Cisco IOS XE CLI • Ncclient 	<ul style="list-style-type: none"> • Cisco 組込イベント マネージャ • Cisco IOS XE CLI
サポートされる rootfs	Busybox、SSH、および Python PIP のインストール	SSH、Yum のインストール、および Python PIP のインストール
GNU C コンパイラ	サポート対象外	サポート対象外
RPM のインストール	サポート対象外	サポート対象
アーキテクチャ	MIPS	x86

ゲストシェルとゲストシェル Lite

ゲストシェルコンテナを使用すると、ユーザは、システム上で自分のスクリプトやアプリケーションを実行できるようになります。Intel x86 プラットフォーム上のゲストシェルコンテナは、CentOS 7.0 の最小限の rootfs を持つ Linux コンテナ (LXC) になります。ランタイム中に、CentOS 7.0 で Yum ユーティリティを使用して、Python バージョン 3.0 などの他の Python ライ

ブラリをインストールすることができます。また、PIPを使用してPythonパッケージをインストールまたは更新することもできます。

Catalyst 3650 や Catalyst 3850 シリーズ スイッチなどの MIPS プラットフォーム上のゲストシェル Lite コンテナには、Carrier Grade Edition (CGE) 7.0 の rootfs があります。ゲストシェル Lite では、スクリプトのインストールまたは実行のみ可能です。これらのデバイスでは、Yum のインストールはサポートされていません。

ゲストシェルのセキュリティ

シスコは、ゲストシェル内のユーザまたはアプリケーションによってホストシステムが攻撃されることがないように、セキュリティを提供しています。ゲストシェルは、ホストカーネルから分離され、非特権コンテナとして動作します。

ゲストシェルのハードウェア要件

この項では、サポート対象のプラットフォームにおけるハードウェア要件に関する情報を提供します。Cisco CSR 1000v と Cisco ISRv (仮想プラットフォーム) は、これらの要件をソフトウェアで実装します。

表 2: Catalyst スイッチでのゲストシェルのサポート

プラットフォーム	デフォルトの DRAM	ゲストシェルのサポート
WS-3650-xxx (すべて)	4 GB	サポート対象
WS-3850-xxx (すべて)	4 GB	サポート対象
C9300-xx-x (すべて)	8 GB	サポート対象
C9500-24Q-x (すべて)	16 GB	サポート対象

Catalyst 3850 シリーズ スイッチの最小システム要件は、4 GB の DRAM です。

表 3: ISR 4000 シリーズ サービス統合型ルータでのゲストシェルのサポート

プラットフォーム	デフォルトの DRAM	ゲストシェルのサポート
ISR 4221	4GB	未サポート
ISR 4321	4 GB	未サポート
	8 GB	サポート対象
ISR 4331	8 GB	サポート対象
	16 GB	サポート対象
ISR 4351	8 GB	サポート対象
	16 GB	サポート対象

プラットフォーム	デフォルトの DRAM	ゲストシェルのサポート
ISR 4431	8 GB	サポート対象
	16 GB	サポート対象
ISR 4451	8 GB	サポート対象
	16 GB	サポート対象

ISR 4000 シリーズ サービス統合型ルータの最小システム要件は、8 GB の DRAM です。



(注) 仮想サービスがインストールされているアプリケーションとゲストシェルコンテナを同時に使用することはできません。

CSR 1000 v と ISRv の最小システム要件は、4 GB の RAM です。

ゲストシェルのストレージ要件

Catalyst 3650 および Catalyst 3850 シリーズスイッチでは、ゲストシェルは、フラッシュのファイルシステムにのみインストールできます。Catalyst 3850 シリーズスイッチのブートフラッシュでは、ゲストシェルを正常にインストールするには 75 MB のディスク空き容量が必要です。

Cisco 4000 シリーズ統合型サービスルータでは、ゲストシェルは、ネットワークインターフェイスモジュール (NIM) のサービスセット識別子 (SSID) (ハードディスク) がある場合、そこにインストールされます。ハードディスクドライブが使用可能な場合、ゲストシェルのインストールにブートフラッシュを選択することはできません。Cisco 4000 シリーズサービス統合型ルータでは、ゲストシェルを正常にインストールするには 1100 MB のハードディスク (NIM SSID) 空き容量が必要です。

Cisco 4000 シリーズ統合型サービスルータおよび ASR 1000 ルータ (オプションのハードディスクがそのルータに追加されている場合) では、ゲストシェルをハードディスクにインストールしており、そのハードディスクがルータに挿入されている場合にのみリソースのサイズ変更を実行できます。



(注) ブートフラッシュを介してインストールしたゲストシェルでは、アプリケーションホスティング設定コマンドを使用したリソースのサイズ変更はできません。

ゲストシェルのインストール中にハードディスク容量が不足した場合、エラーメッセージが表示されます。

次に、ISR 4000 シリーズルータでのエラーメッセージの例を示します。

```
% Error:guestshell_setup.sh returned error:255, message:
```

Not enough storage for installing guestshell. Need 1100 MB free space.

ブートフラッシュまたはハードディスクの空き領域は、ゲストシェルが追加データを格納するために使用されることがあります。Cisco Catalyst 3850 シリーズスイッチでは、ゲストシェルが使用できるストレージ容量は 18 MB です。Cisco 4000 シリーズ サービス統合型ルータでは、ゲストシェルが使用できるストレージ容量は 800 MB です。ゲストシェルはブートフラッシュにアクセスするため、その空き領域の全体を使用できます。

表 4: ゲストシェルおよびゲストシェル *Lite* が使用できるリソース

リソース	デフォルト	最小/最大
CPU	1 % (注) 1%は非標準。800 CPU ユニット/システム CPU ユニットの全体	1/100 %
メモリ	256 MB 512 MB (Cisco CSR 1000v)	256/256 MB 512/512 MB (Cisco CSR 1000v)

デバイスでのゲストシェルへのアクセス

ネットワーク管理者は、IOS コマンドを使用して、ゲストシェル内のファイルおよびユーティリティを管理することができます。

ゲストシェルのインストール中に、SSH アクセスがキーベースの認証でセットアップされます。ゲストシェルへのアクセスは、IOS の最も高い特権 (15) を持つユーザに制限されます。このユーザは、`sudo` の実行者である `guestshell Linux` ユーザとして Linux コンテナへのアクセスを許可され、すべてのルート操作を実行できます。ゲストシェルから実行されるコマンドは、ユーザが IOS 端末にログインしたときと同じ特権で実行されます。

ゲストシェルプロンプトでは、標準的な Linux コマンドを実行できます。

管理ポートを介してのゲストシェルへのアクセス

ゲストシェルは、デフォルトで、アプリケーションによる管理ネットワークへのアクセスを許可します。ユーザは、ゲストシェル内から管理 VRF のネットワーク設定を変更することはできません。



- (注) 管理ポートがないプラットフォームの場合、`VirtualPortGroup` を IOS 設定内のゲストシェルに関連付けることができます。詳細については、「`VirtualPortGroup` の設定例」の項を参照してください。

ゲスト シェルでのスタッキング

ゲストシェルがインストールされている場合、フラッシュのファイルシステムには、ディレクトリが自動的に作成されます。このディレクトリは、スタックメンバー間で同期されます。切り替え時には、このディレクトリの内容のみが、すべてのスタックメンバー間で同期されます。ハイ アベイラビリティでの切り替えの際にデータを保持するには、このディレクトリにデータを格納します。

ハイアベイラビリティでの切り替えの際には、新しいアクティブデバイスは、それぞれのゲストシェルインストールを作成し、ゲストシェルを同期状態に復元します。古いファイルシステムは維持されません。ゲストシェルの状態は、すべてのスタックメンバー間で内部的に同期されます。

IOx の概要

IOx は Cisco が開発したエンド ツー エンド アプリケーション フレームワークであり、Cisco ネットワーク プラットフォーム上のさまざまなタイプのアプリケーションに対し、アプリケーション ホスティング 機能を提供します。Cisco ゲストシェルは特殊なコンテナ展開であり、システムの開発および使用に役立つアプリケーションの 1 つです。

IOx は、構築済みアプリケーションをパッケージ化し、それらをターゲットデバイス上にホストする開発者の作業を支援する一連のサービスを提供することにより、アプリケーションのライフサイクル管理とデータ交換を容易化します。IOx のライフサイクル管理には、アプリケーションおよびデータの配布、展開、ホスティング、開始、停止（管理）、およびモニターが含まれます。IOx サービスにはアプリケーションの配布および管理ツールも含まれており、ユーザーがアプリケーションを発見して IOx フレームワークに展開するのに役立ちます。

アプリケーション ホスティングは、次の機能を提供します。

- ネットワークの不均質性の遮蔽。
- デバイス上にホストされているアプリケーションのライフサイクルをリモートで管理する IOx アプリケーション プログラミング インターフェイス (API) 。
- 一元的なアプリケーション ライフ サイクル管理。
- クラウド ベースの開発。

IOx のトレースとロギングの概要

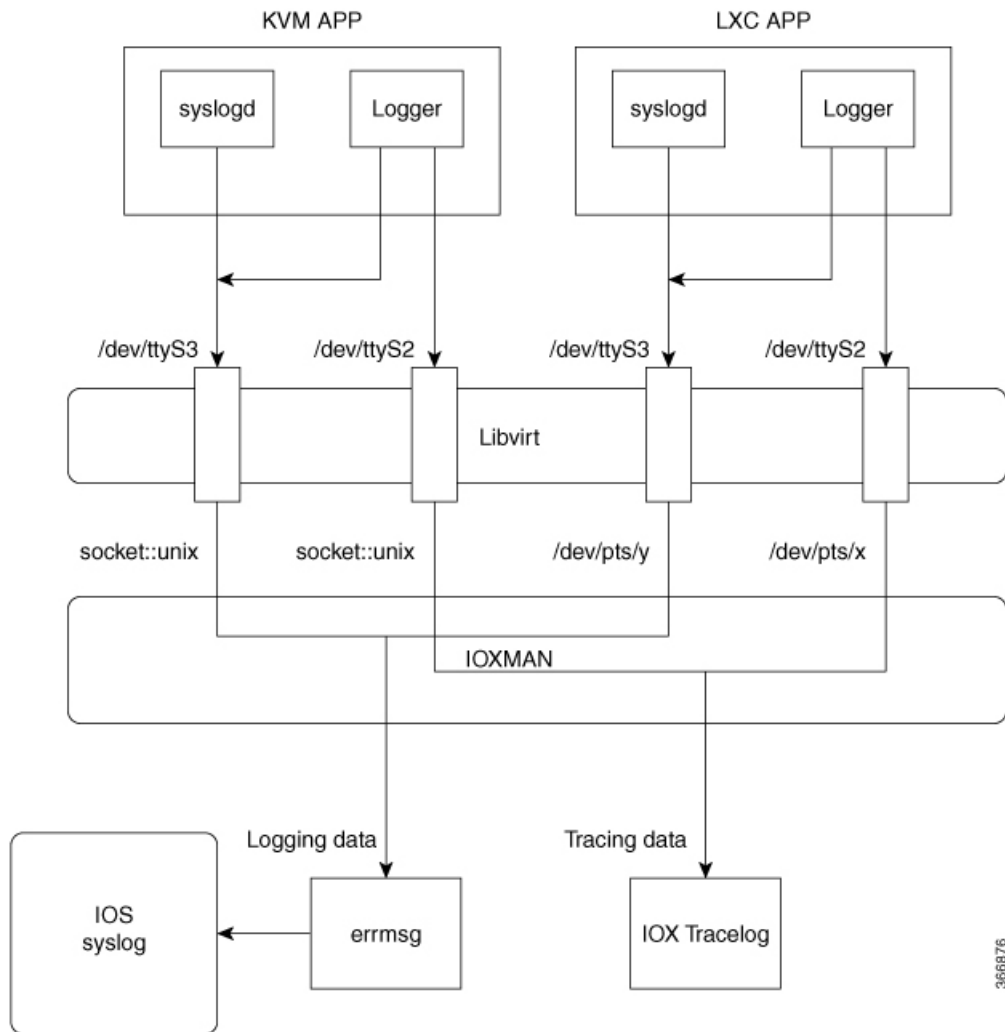
IOx のトレースとロギングの機能を使用すると、ホスト デバイスでゲストアプリケーションを個別に実行できます。これにより、ホストへのデータのロギングとトレースをレポートするのに役立ちます。トレース データは IOx トレースログに保存され、ロギング データはホスト デバイスの IOS syslog に保存されます。

トレース データをホスト デバイス上の適切なストレージ デバイスにリダイレクトすると、ゲストアプリケーションのデバッグに役立ちます。

IOXMAN 構造体

ゲストアプリケーション、システム LXC、または KVM インスタンスはそれぞれ独自の syslog およびログファイルを使用して設定されます。これらのファイルは表示可能なファイルシステム内に保存され、ホストデバイスからはアクセスできません。IOS syslog へのデータのロギングとホスト上の IOx トレース ログへのデータのトレースをサポートするため、次の図に示すように、ホストにデータを配信するための2つのシリアルデバイス (`/dev/ttyS2` と `/dev/ttyS3`) がゲストアプリケーションで指定されています。

図 1: IOXMAN 構造体



IOXMAN は、トレース インフラストラクチャを確立してロギング サービスまたはトレース サービス (シリアルデバイスをエミュレートする Libvirt を除く) を提供するプロセスです。IOXMAN は、ゲストアプリケーションのライフサイクルに基づいて、トレース サービスを有効または無効にし、ロギングデータを IOS syslog に送信し、トレースデータを IOx トレース ログに保存し、各ゲストアプリケーションの IOx トレース ログを維持します。

ロギングとトレースのシステム フロー

ここでは、IOx のロギングとトレースの仕組みについて説明します。

LXC のロギング

1. ゲスト OS が、ゲスト アプリケーションで `/dev/ttyS2` を有効にします。
2. ゲスト アプリケーションが、`/dev/ttyS2` にデータを書き込みます。
3. Libvirt が、ホストで `/dev/pts/x` への `/dev/ttyS2` をエミュレートします。
4. IOXMAN が、エミュレートされたシリアルデバイス `/dev/pts/x` を XML ファイルから取得します。
5. IOXMAN が、使用可能なデータを `/dev/pts/x` からリッスンして読み取り、メッセージの重大度を設定して、メッセージをフィルタ処理し、解析してキューに格納します。
6. `errmsg` を使用してホストの `/dev/log` デバイスにメッセージを送信するタイマーが開始されます。
7. データが IOS syslog に保存されます。

KVM のロギング

1. ゲスト OS が、ゲスト アプリケーションで `/dev/ttyS2` を有効にします。
2. ゲスト アプリケーションが、`/dev/ttyS2` にデータを書き込みます。
3. Libvirt が、ホストで `/dev/pts/x` への `/dev/ttyS2` をエミュレートします。
4. IOXMAN が、エミュレートされた TCP パスを XML ファイルから取得します。
5. IOXMAN が、UNIX ソケットを開き、リモート ソケットに接続します。
6. IOXMAN が、使用可能なデータをソケットから読み取り、メッセージの重大度を設定して、メッセージをフィルタ処理し、解析して、キューに格納します。
7. `errmsg` を使用してホストの `/dev/log` デバイスにメッセージを送信するタイマーが開始されます。
8. データが IOS syslog に保存されます。

LXC のトレース

1. ゲスト OS が、ゲスト アプリケーションで `/dev/ttyS3` を有効にします。
2. メッセージを `/dev/ttyS3` にコピーするように `syslogd` を設定します。
3. ゲスト アプリケーションが、`/dev/ttyS3` にデータを書き込みます。
4. Libvirt が、ホストで `/dev/pts/y` への `/dev/ttyS3` をエミュレートします。

5. IOXMAN が、エミュレートされたシリアル デバイス `/dev/pts/y` を XML ファイルから取得します。
6. IOXMAN が、使用可能なデータを `/dev/pts/y` からリッスンして読み取り、フィルタ処理し、解析して、メッセージを IOx トレースログに保存します。
7. IOx トレースログが満杯の場合は、IOXMAN がトレースログファイルを `/bootflash/tracelogs` にローテーションします。

KVM のトレース

1. ゲスト OS が、ゲスト アプリケーションで `/dev/ttyS3` を有効にします。
2. メッセージを `/dev/ttyS3` にコピーするように `syslogd` を設定します。
3. ゲスト アプリケーションが、`/dev/ttyS3` にデータを書き込みます。
4. Libvirt が、ホストで TCP パスへの `/dev/ttyS3` をエミュレートします。
5. IOXMAN が、エミュレートされた TCP パスを XML ファイルから取得します。
6. IOXMAN が、UNIX ソケットを開き、リモート ソケットに接続します。
7. IOXMAN が、使用可能なデータをソケットから読み取り、メッセージの重大度を設定して、メッセージをフィルタ処理し、解析して、IOx トレースログに格納します。
8. IOx トレースログが満杯の場合は、IOXMAN がトレースログファイルを `/bootflash/tracelogs` にローテーションします。

メッセージのロギングとトレース

ここでは、IOS syslog へのメッセージのロギングとトレースについて説明します。

IOS syslog でのメッセージのロギング

ゲスト アプリケーションから受信したどのロギング メッセージでも、IOXMAN はメッセージの重大度をデフォルトで NOTICE に設定してから IOS syslog に送信します。IOSd で受信されたメッセージはコンソールに表示され、次のメッセージ形式で IOS syslog に保存されます。

```
*Apr 7 00:48:21.911: %IM-5-IOX_INST_NOTICE:ioxman: IOX SERVICE guestshell LOG: Guestshell test
```

IOS syslog に準拠するために、IOXMAN はロギング メッセージの重大度をサポートしています。重大度のあるロギング メッセージを報告するには、ゲスト アプリケーションでメッセージの先頭にヘッダーを追加する必要があります。

```
[a123b234,version,severity]
```

```
a123b234 is magic number.
Version:          severity support version.  Current version is 1.
Severity:         CRIT is 2
                  ERR is 3
                  WARN is 4
                  NOTICE is 5
```

```
INFO is 6
DEBUG is 7
```

次に、メッセージ ログの例を示します。

```
echo "[a123b234,1,2]Guestshell failed" > /dev/ttyS2
```

ゲストアプリケーションから IOS syslog にロギング データを報告するには、次の手順を実行します。

1. Cプログラミングを使用している場合は、**write()** を使用してロギングデータをホストに送信します。

```
#define SYSLOG_TEST      "syslog test"
int fd;
fd = open("/dev/ttyS2", O_WRONLY);
write(fd, SYSLOG_TEST, strlen(SYSLOG_TEST));
close(fd);
```

2. シェル コンソールを使用している場合は、**echo** を使用してロギングデータをホストに送信します。

```
echo "syslog test" > /dev/ttyS2
```

IOx トレースログへのメッセージのトレース

ゲストアプリケーションから IOx トレースログにトレース メッセージを報告するには、次の手順を実行します。

1. Cプログラミングを使用している場合は、**write()** を使用してトレースメッセージをホストに送信します。

```
#define SYSLOG_TEST      "tracelog test"
int fd;
fd = open("/dev/ttyS3", O_WRONLY);
write(fd, SYSLOG_TEST, strlen(SYSLOG_TEST));
close(fd);
```

2. Cプログラミングを使用している場合は、**syslog()** を使用してトレースメッセージをホストに送信します。

```
#define SYSLOG_TEST      "tracelog test"

syslog(LOG_INFO, "%s\n", SYSLOG_TEST);
```

3. シェル コンソールを使用している場合は、**echo** を使用してトレースデータをホストに送信します。

```
echo "tracelog test" > /dev/ttyS3
or
logger "tracelog test"
```

例：ゲストシェルのネットワーキング設定

ゲストシェルのネットワーキングでは、次の設定が必要です。

- ドメインネームシステム (DNS) の設定
- プロキシの設定
- プロキシの設定を使用するための YUM または PIP の設定

ゲストシェルを有効にする方法

IOx の管理

始める前に

IOxは開始まで最長で2分かかります。ゲストシェルを正常に有効にするには、CAF、IOXman、および Libirtd サービスが実行している必要があります。

手順の概要

1. **enable**
2. **configure terminal**
3. **iox**
4. **exit**
5. **show iox-service**
6. **show app-hosting list**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	enable 例： Device> enable	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	configure terminal 例： Device# configure terminal	グローバル コンフィギュレーション モードを開始します。
ステップ 3	iox 例： Device(config)# iox	IOx サービスを設定します。

	コマンドまたはアクション	目的
ステップ 4	exit 例： Device(config)# exit	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 5	show iox-service 例： Device# show iox-service	IOx サービスのステータスを表示します。
ステップ 6	show app-hosting list 例： Device# show app-hosting list	デバイスに対して有効になっている app-hosting サービスのリストを表示します。

次のタスク

次に、ISR 4000 シリーズ ルータでの **show iox-service** コマンドの出力例を示します。

```
Device# show iox-service

Virtual Service Global State and Virtualization Limits:

Infrastructure version : 1.7
Total virtual services installed : 0
Total virtual services activated : 0

Machine types supported   : KVM, LXC
Machine types disabled   : none

Maximum VCPUs per virtual service : 6
Resource virtualization limits:
Name                       Quota      Committed  Available
-----
system CPU (%)              75         0           75
memory (MB)                 10240     0           10240
bootflash (MB)              1000      0           1000
harddisk (MB)                20000     0           18109
volume-group (MB)           190768    0           170288

IOx Infrastructure Summary:
-----
IOx service (CAF)          : Running
IOx service (HA)           : Not Running
IOx service (IOxman)      : Running
Libvirtd                   : Running
```

次に示すのは、Catalyst 3850 シリーズ スイッチでの **show iox-service** コマンドの短縮された出力例です。

```
Device# show iox-service

IOx Infrastructure Summary:
-----
IOx service (CAF)          : Running
```

```
IOx service (HA)      : Running
IOx service (IOxman) : Running
Libvirtd             : Running
```

次に、**show app-hosting list** コマンドの出力例を示します。

```
Device# show app-hosting list

App id                               State
-----
guestshell                            RUNNING
```

ゲストシェルの管理



(注) VirtualPortGroups はルーティングプラットフォームでのみサポートされています。

始める前に

ゲストシェルのアクセスが機能するには、IOx が構成されて実行している必要があります。IOx が構成されていない場合は、IOx の構成を求めるメッセージが表示されます。IOx を削除すると、ゲストシェルにもアクセスできなくなります。ただし rootfs は影響を受けません。

ゲストシェルを有効にして操作するように、アプリケーションまたは管理インターフェイスも設定する必要があります。ゲストシェルのインターフェイスを有効にする方法の詳細については、「Configuring the AppGigabitEthernet Interface for Guest Shell」および「Enabling Guest Shell on the Management Interface」のセクションを参照してください。

手順の概要

1. **enable**
2. **guestshell enable**
3. **guestshell run linux-executable**
4. **guestshell run bash**
5. **guestshell disable**
6. **guestshell destroy**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	enable 例： Device> enable	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	guestshell enable 例：	ゲストシェルサービスの有効化。

	コマンドまたはアクション	目的
	Device# <code>guestshell enable</code>	<p>(注)</p> <ul style="list-style-type: none"> • guestshell enable コマンドは、ネットワークに管理 Virtual Routing and Forwarding (VRF) インスタンスを使用します。 • フロントパネル ネットワーキングに VirtualPortGroups (VPG) を使用している場合は、まず VPG を構成する必要があります。 • ゲスト IP アドレスとゲートウェイ IP アドレスは同じサブネット内にある必要があります。
ステップ 3	<p>guestshell run linux-executable</p> <p>例 :</p> <pre>Device# guestshell run python</pre> <p>または</p> <pre>Device# guestshell run python3</pre>	<p>ゲスト シェルで Linux プログラムを実行します。</p> <p>(注) Cisco IOS XE Amsterdam 17.3.1 以降のリリースでは、Python バージョン 3 のみがサポートされます。</p>
ステップ 4	<p>guestshell run bash</p> <p>例 :</p> <pre>Device# guestshell run bash</pre>	<p>Bash シェルを開始して、ゲスト シェルにアクセスします。</p>
ステップ 5	<p>guestshell disable</p> <p>例 :</p> <pre>Device# guestshell disable</pre>	<p>ゲスト シェル サービスを無効化します。</p>
ステップ 6	<p>guestshell destroy</p> <p>例 :</p> <pre>Device# guestshell destroy</pre>	<p>ゲスト シェル サービスを非アクティブ化して、アンインストールします。</p>

ゲストシェルの有効化と実行

guestshell enable コマンドは、ゲスト シェルをインストールします。このコマンドは、無効化されているゲスト シェルを再アクティブ化する際にも使用されます。

ゲスト シェルが有効化された状態でシステムをリロードすると、ゲスト シェルは有効化されたままになります。



(注) **guestshell enable** コマンドを使用する前に、IOx を設定しておく必要があります。

guestshell run bash コマンドは、ゲストシェルの **bash** プロンプトを開きます。このコマンドを動作させるには、ゲストシェルが事前に有効化されていることが必要です。



- (注) 次のメッセージがコンソールに表示される場合、IOx が有効化されていません。 **show iox-service** コマンドの出力をチェックして、IOx の状態を確認してください。

```
The process for the command is not responding or is otherwise unavailable
```

ゲストシェルを有効にする方法の詳細については、「[Configuring the AppGigabitEthernet Interface for Guest Shell](#)」および「[Enabling Guest Shell on the Management Interface](#)」のセクションを参照してください。

ゲストシェルの無効化と破棄

guestshell disable コマンドを使用することで、ゲストシェルを終了して無効化できます。ゲストシェルが無効化された状態でシステムをリロードすると、ゲストシェルは無効化されたままになります。

guestshell destroy コマンドは、フラッシュのファイルシステムから **rootfs** を削除します。すべてのファイル、データ、インストールされている Linux アプリケーション、およびカスタムの Python ツールとユーティリティが削除され、回復できなくなります。

アプリケーションホスティングを使用したゲストシェルの管理



- (注) この項は、シスコルーティングプラットフォームに適用されます。VirtualPortGroups は、Cisco Catalyst スイッチングプラットフォームではサポートされていません。

ゲストシェルのアクセスが機能するには、IOx が構成されて実行している必要があります。IOx が構成されていない場合は、IOx の構成を求めるメッセージが表示されます。IOx を削除すると、ゲストシェルにもアクセスできなくなります。ただし **rootfs** は影響を受けません。



- (注) この手順（アプリケーションホスティングを使用したゲストシェルの管理）を使用して、Cisco IOS XE Fuji 16.7.1 以降のリリースのゲストシェルを有効にします。Cisco IOS XE Everest 16.6.x 以前では、[ゲストシェルの管理（13 ページ）](#) の手順を使用します。

```
Device(config)# interface GigabitEthernet1
Device(config-if)# ip address dhcp
Device(config-if)# ip nat outside
Device(config-if)# exit

Device(config-if)# interface VirtualPortGroup0
Device(config-if)# ip address 192.168.35.1 255.255.255.0
Device(config-if)# ip nat inside
```

```

Device(config-if)# exit

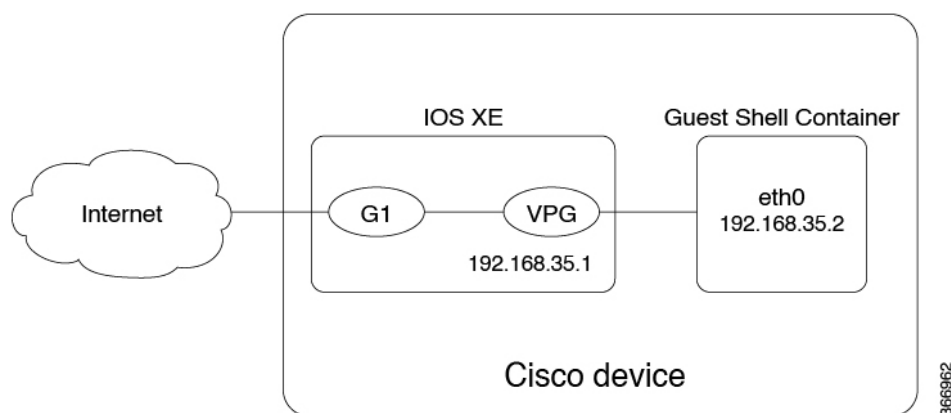
Device(config)# ip nat inside source list GS_NAT_ACL interface GigabitEthernet1 overload
Device(config)# ip access-list standard GS_NAT_ACL
Device(config)# permit 192.168.0.0 0.0.255.255

Device(config)# app-hosting appid guestshell
Device(config-app-hosting)# app-vnic gateway1 virtualportgroup 0 guest-interface 0
guest-ipaddress 192.168.35.2
netmask 255.255.255.0 gateway 192.168.35.1 name-server 8.8.8.8 default
Device(config-app-hosting)# app-resource profile custom cpu 1500 memory 512
Device(config-app-hosting)# end

Device# guestshell enable
Device# guestshell run python

```

図 2: アプリケーション ホスティングを使用したゲストシェルの管理



前面パネルのネットワーキングでは、GigabitEthernet インターフェイスと VirtualPortGroup インターフェイスを上図に示すように設定する必要があります。ゲストシェルは Virtualportgroup を送信元インターフェイスとして使用し、NAT を通じて外部ネットワークに接続します。

内部 NAT の設定には、次のコマンドを使用します。これにより、ゲストシェルがインターネットに到達し、たとえば、Linux ソフトウェア更新プログラムを取得できるようになります。

```

ip nat inside source list
ip access-list standard
permit

```

上の例の **guestshell run** コマンドは Python 実行可能ファイルを実行します。また、**guestshell run** コマンドを使用して他の Linux 実行可能ファイルを実行することもできます。たとえば、**guestshell run bash** コマンドは bash シェルを起動し、**guestshell disable** コマンドはゲストシェルのシャットダウンして無効にします。後でシステムをリロードしても、ゲストシェルは無効のままになります。

Python インタープリタのアクセス

Python はインタラクティブに使用できますが、Python スクリプトをゲストシェルで実行することもできます。**guestshell run python** コマンドを使用してゲストシェルで Python インタープリタを起動し、Python 端末を開きます。



- (注) Cisco IOS XE Amsterdam 17.3.1 より前のリリースでは、Python V2 がデフォルトです。Cisco IOS XE Amsterdam 17.1.1 および Cisco IOS XE Amsterdam 17.2.1 では、Python V3 がサポートされています。Cisco IOS XE Amsterdam 17.3.1 以降のリリースでは、Python V3 がデフォルトです。

Cisco IOS XE Amsterdam 17.3.1 より前のリリース

guestshell run コマンドは、Linux 実行可能ファイルの実行に相当する Cisco IOS であり、Cisco IOS からの Python スクリプトの実行時に絶対パスを指定します。次の例は、コマンドの絶対パスを指定する方法を示しています。

```
Guestshell run python /flash/guest-share/sample_script.py parameter1 parameter2
```

次に、Cisco Catalyst 3650 シリーズ スイッチまたは Cisco Catalyst 3850 シリーズ スイッチで Python を有効にする例を示します。

```
Device# guestshell run python

Python 2.7.11 (default, March 16 2017, 16:50:55)
[GCC 4.7.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>>
```

次の例は、Cisco ISR 4000 シリーズ サービス統合型ルータで Python を有効にする方法を示しています。

```
Device# guestshell run python

Python 2.7.5 (default, Jun 17 2014, 18:11:42)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>>
```

Cisco IOS XE Amsterdam 17.3.1 以降のリリース

次の例は、Cisco Catalyst 9000 シリーズ スイッチ上で Python を有効にする方法を示しています。

```
Device# guestshell run python3

Python 3.6.8 (default, Nov 21 2019, 22:10:21)
[GCC 8.3.1 20190507 (Red Hat 8.3.1-4)] on linux
Type "help", "copyright", "credits" or "license" for more information.>>>>
```

ゲストシェルの設定例

例：ゲストシェルの管理

次の例では、Catalyst 3850 シリーズ スイッチ上でゲストシェルを有効にする方法を示しています。

```
Device> enable
Device# guestshell enable

Management Interface will be selected if configured
Please wait for completion
Guestshell enabled successfully

Device# guestshell run python

Python 2.7.11 (default, Feb 21 2017, 03:39:40)
[GCC 5.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.

Device# guestshell run bash

[guestshell@guestshell ~]$

Device# guestshell disable

Guestshell disabled successfully

Device# guestshell destroy

Guestshell destroyed successfully
```

VirtualPortGroup 設定の例



(注) VirtualPortGroups は Cisco ルーティング プラットフォームでのみサポートされています。

ゲストシェルネットワークングに VirtualPortGroup インターフェイスを使用する場合、VirtualPortGroup インターフェイスには設定済みの静的 IP アドレスが必要です。フロントポートインターフェイスはインターネットに接続されている必要があり、ネットワーク アドレス変換 (NAT) は VirtualPortGroup とフロント パネル ポートの間で設定されている必要があります。

次に示すのは、VirtualPortGroup の設定例です。

```
Device> enable
Device# configure terminal
Device(config)# interface VirtualPortGroup 0
Device(config-if)# ip address 192.168.35.1 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# no mop enabled
Device(config-if)# no mop sysid
Device(config-if)# exit
Device(config)# interface GigabitEthernet 0/0/3
Device(config-if)# ip address 10.0.12.19 255.255.0.0
Device(config-if)# ip nat outside
Device(config-if)# negotiation auto
Device(config-if)# exit
Device(config)# ip route 0.0.0.0 0.0.0.0 10.0.0.1
Device(config)# ip route 10.0.0.0 255.0.0.0 10.0.0.1
!Port forwarding to use ports for SSH and so on.
Device(config)# ip nat inside source static tcp 192.168.35.2 7023 10.0.12.19 7023
extendable
Device(config)# ip nat outside source list NAT_ACL interface GigabitEthernet 0/0/3
overload
Device(config)# ip access-list standard NAT_ACL
Device(config-std-nacl)# permit 192.168.0.0 0.0.255.255
Device(config-std-nacl)# exit

! App-hosting configuration
Device(config)# app-hosting appid guestshell
Device(config-app-hosting)# app-vnic gateway1 virtualportgroup 0 guest-interface 0
Device(config-app-hosting-gateway)# guest-ipaddress 192.168.35.2 netmask 255.255.255.0
Device(config-app-hosting-gateway)# exit
Device(config-app-hosting)# app-resource profile custom
Device(config-app-resource-profile-custom)# cpu 1500
Device(config-app-resource-profile-custom)# memory 512
Device(config-app-resource-profile-custom)# end

Device# guestshell enable
Device# guestshell run python
```

例：ゲストシェルの使用

ゲストシェルプロンプトから Linux のコマンドを実行できます。次の例は、一部の Linux コマンドの使用法を示しています。

```
[guestshell@guestshell~]$ pwd
/home/guestshell

[guestshell@guestshell~]$ whoami
guestshell

[guestshell@guestshell~]$ uname -a
Linux guestshell 3.10.101.cge-rt110 #1 SMP Sat Feb 11 00:33:02
PST 2017 mips64 GNU/Linux
```

Catalyst 3650 および Catalyst 3850 シリーズ スイッチには、BusyBox が提供する定義された一連の Linux 実行可能ファイルがあり、Cisco 4000 シリーズ サービス統合型ルータには、CentOS Linux リリース 7.1.1503 が提供するコマンドがあります。

次の例は、Catalyst 3850 シリーズ スイッチ上での **dohost** コマンドの使用を示しています。

```
[guestshell@guestshell ~]$ dohost "show version"

Cisco IOS Software [Everest], Catalyst L3 Switch Software [CAT3K_CAA-UNIVERSALK9-M],
Experimental Version 16.5.2017200014[v165_throttle-BLD-
BLD_V165_THROTTLE_LATEST_20170531_192849_132]
```



(注) **dohost** コマンドには、**ip http server** コマンドがデバイス上で設定されていることが必要です。

例：ゲストシェルのネットワーキング設定

ゲストシェルのネットワーキングでは、次の設定が必要です。

- ドメイン ネーム システム (DNS) の設定
- プロキシの設定
- プロキシの設定を使用するための YUM または PIP の設定

ゲストシェルの DNS 設定の例

ゲストシェルのサンプル DNS 構成は次のとおりです。

```
[guestshell@guestshell ~]$ cat/etc/resolv.conf
nameserver 192.0.2.1

Other Options:
[guestshell@guestshell ~]$ cat/etc/resolv.conf
domain cisco.com
search cisco.com
nameserver 192.0.2.1
search cisco.com
nameserver 198.51.100.1
nameserver 172.16.0.6
domain cisco.com
nameserver 192.0.2.1
nameserver 172.16.0.6
nameserver 192.168.255.254
```

例：プロキシ環境変数の設定

ネットワークがプロキシの背後にある場合は、Linux でプロキシ変数を設定します。必要な場合は、環境にこれらの変数を追加します。

次の例は、プロキシ変数を設定する方法を示しています。

```
[guestshell@guestshell ~]$cat /bootflash/proxy_vars.sh
export http_proxy=http://proxy.example.com:80/
export https_proxy=http://proxy.example.com:80/
export ftp_proxy=http://proxy.example.com:80/
export no_proxy=example.com
export HTTP_PROXY=http://proxy.example.com:80/
export HTTPS_PROXY=http://proxy.example.com:80/
export FTP_PROXY=http://proxy.example.com:80/
guestshell ~] source /bootflash/proxy_vars.sh
```

例：プロキシ設定用の Yum および PIP の構成

次の例は、プロキシ環境変数の設定に Yum を使用する方法を示しています。

```
cat /etc/yum.conf | grep proxy
[guestshell@guestshell~]$ cat/bootflash/yum.conf | grep proxy
proxy=http://proxy.example.com:80/
```

PIP のインストールでは、プロキシ設定に使用される環境変数が選択されます。PIP インストールには `-E` オプションを指定した `sudo` を使用します。環境変数が設定されていない場合は、次の例に示すように PIP コマンドでそれらを明示的に定義します。

```
sudo pip --proxy http://proxy.example.com:80/install requests
sudo pip install --trusted-host pypi.example.com --index-url
http://pypi.example.com/simple requests
```

次の例では、Python の PIP インストールを使用する方法を示します。

```
Sudo -E pip install requests
[guestshell@guestshell ~]$ python
Python 2.17.11 (default, Feb 3 2017, 19:43:44)
[GCC 4.7.0] on linux2
Type "help", "copyright", "credits" or "license" for more information
>>>import requests
```

ゲスト シェルに関するその他の参考資料

関連資料

関連項目	マニュアル タイトル
	『Programmability Command Reference, Cisco IOS XE Everest 16.6.1』
Python モジュール	『CLI Python モジュール』
ゼロ タッチ プロビジョニング	『ゼロ タッチ プロビジョニング』

MIB

MB	MIB のリンク
	<p>選択したプラットフォーム、Cisco IOS リリース、およびフィチャセットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。</p> <p>http://www.cisco.com/go/mibs</p>

シスコのテクニカル サポート

説明	リンク
<p>シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンラインリソースを提供しています。</p> <p>お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service (Field Notice からアクセス)、Cisco Technical Services Newsletter、Really Simple Syndication (RSS) フィードなどの各種サービスに加入できます。</p> <p>シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。</p>	<p>http://www.cisco.com/support</p>

ゲスト シェルの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェア イメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 5: ゲストシェルの機能情報

機能名	リリース	機能情報
<p>ゲスト シェル</p>	<p>Cisco IOS XE Everest 16.5.1a Cisco IOS XE Everest 16.5.1b</p>	<p>ゲストシェルは、お客様がシスコスイッチの自動制御および管理のためのカスタム Python アプリケーションを実行できる、埋め込み Linux 環境であるセキュア コンテナです。システムの自動化されたプロビジョニングも含まれます。このコンテナシェルは、ホストデバイスから分離された安全な環境を提供します。ユーザはそこで、スクリプトまたはソフトウェアパッケージをインストールし、実行することができます。</p> <p>Cisco IOS XE Everest 16.5.1a では、この機能は次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none"> • Cisco Catalyst 3650 シリーズスイッチ • Cisco Catalyst 3850 シリーズスイッチ • Cisco Catalyst 9300 シリーズスイッチ • Cisco Catalyst 9500 シリーズスイッチ <p>Cisco IOS Everest 16.5.1b では、この機能は次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none"> • Cisco 4000 シリーズ サービス統合型ルータ
	<p>Cisco IOS XE Everest 16.6.2</p>	<p>この機能は、Cisco IOS XE Everest 16.6.2 で、Cisco Catalyst 9400 シリーズスイッチに実装されました。</p>

機能名	リリース	機能情報
	Cisco IOS XE Fuji 16.7.1	<p>Cisco IOS XE Fuji 16.7.1 では、この機能は次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none"> • Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ • Cisco Cloud Services Router 1000V シリーズ <p>Cisco IOS XE Fuji 16.7.1 では、ゲストシェル機能の場合、ロギングとトレーシングサポートが Cisco ASR 1000 アグリゲーションサービスルータに実装されました。</p>
	Cisco IOS XE Fuji 16.8.1	<p>Cisco IOS XE Fuji 16.8.1 では、この機能は Cisco Catalyst 9500 ハイパフォーマンスシリーズスイッチに実装されていました。</p>
	Cisco IOS XE Fuji 16.9.1	<p>Cisco IOS XE Fuji 16.9.1 では、この機能は Cisco 1000 シリーズ サービス統合型ルータに実装されていました。</p>

翻訳について

このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。